IBM® Netezza® Analytics
Release 2.0.1

IBM Netezza Analytics Release Notes

Part Number 00J2008-03 Rev. 2

Note: Before using this information and the product that it supports, read the information in “Notices and Trademarks” on page 29.

Contents

General IBM Netezza Analytics Topics

IMPORTANT – Read First Before Installing This Release

Database Compatibility

This release of the IBM Netezza Analytics (referred to as Netezza Analytics in the remainder of this document) supports Netezza systems that run release **6.0.5P5** or later. If your Netezza system is using an earlier release, you must upgrade it before using this release of Netezza Analytics.

Compatibility With Prior Releases

Release 2.0 introduced some changes to algorithms that are not backward compatible or that change the behavior of the algorithm. If you have applications that call Netezza Analytics algorithms, you may need to change the algorithm calls in cases where the algorithm's parameters have changed, the default behavior changed, or the algorithm name changed. For more information on these backward compatibility issues see the following sections:

- Call Interface Changes to Analytic Functions
- FPGROWTH Algorithms Renamed and Modified
- Changes to the KMEANS Algorithm
- Changes to nzMatrix

Compatibility With Revolution R Enterprise for IBM Netezza

This release of Netezza Analytics requires Revolution R Enterprise for IBM Netezza release **5.0.3** or greater.

Appliance Workload Impact

Netezza Analytics functionality is broad and powerful, and as such, utilizes various resources on the appliance. In some instances the resources required to perform certain analytic operations could be substantial (for example, large matrix calculations). Many safeguards are in place to ensure continued performance of the overall appliance, but it should be noted that when performing complex operations there could be an impact on other “normal” systems-related activity. If you experience performance differences on your “normal” systems, take note of what, if any, analytic operations are simultaneously being performed. In the future you can run these operations at a time that does not impact critical operations or you can adjust settings to run the analytic operations with a lower priority.

New Features in Netezza Analytics Release 2.0

Call Interface Changes to Analytic Functions

The call interface to many analytic functions has changed in this release. The modifications were implemented to provide uniformity and consistency between all algorithms and also to accomplish the following goals:

- Support for definition of nominal and numeric attributes, providing uniform processing for different data types
- Support for column properties
- Support for “check mode” facility when calling algorithms
- Support for handling missing values (see the new feature description Missing Value Support in Analytics Functions)
- Standardized event messages and parameter validation

In the calls to analytic stored procedures, changes fall into the following categories:

- Elimination of specific named parameters that identify column(s) to be used by the algorithm (in various roles). Instead input columns are provided in the '*incolumn*' parameter.
- Addition of 'roles' associated with columns
- Situations where '*outtable*' or other named parameter has been replaced by '*model*' in cases where the output from an algorithm is a model.
- Combining multiple values for '*class*', '*class1*', '*class2*' parameters into one '*class*' parameter.

Backward Compatibility

Not all algorithms have backward compatibility with the changed call interface. For those that do not (listed in the table below), you must change your calls. It is highly recommended that you also change calls for those algorithms that do provide backward compatibility, since in the future this backward compatibility will be deprecated.

Call Interface Changes to Analytic Functions

The table below lists functions with a changed call interface and provides examples of the changes through sample calls. Only the changed portion of the parameter list is shown; the ellipse (…) in the sample call assumes that other required parameters are provided and have not changed from an older release to this release. The table also notes whether backward compatibility is available:

Prior Release (old way) (Sample partial call)	**Release 2.0 (new way)** (Sample partial call)	**Backward Compatible? /Notes**
call nza..ANOVA_CRD_TEST('**variable=yield**, **...**');	call nza..ANOVA_CRD_TEST('**incolumn=yield**, **…** ');	No
call nza..ANOVA_RBD_TEST('**variable=val**, **...**');	call nza..ANOVA_RBD_TEST('**incolumn=val**, **...**');	No
call nza..BITABLE('**incolumn1=income; incolumn2=education ...**')	call nza..BITABLE(' '**incolumn=income:x; education:y, ...**')	No/Changed in Release 2.0.1
call nza..CANONICAL_CORR('**X=residualsugar; acidity, Y=density; pH, ...**');	call nza..CANONICAL_CORR ('**incolumn=residualsugar:X; acidity:X; density:Y; pH:Y**, **...**');	No
call nza..CHISQ_TEST('**X=race, Y=sex, ...**');	call nza..CHISQ_TEST('**incolumn=race; sex**, **...**');	No
call nza..COND_ENTROPY ('**X=age, Y=wage_per_hour**, **...**);	call nza..COND_ENTROPY('**incolumn=age:X; wage_per_hour:Y**, **...**');	No
call nza..CORR('**X=age, Y=wage_per_hour**, **...**');	call nza..CORR('**incolumn=age; wage_per_hour**, **...**');	No
call nza..CORRELATION1000MATRIX('**x=sepallength;sepalwidth, y=petallength;petalwidth**, **...**');	call nza..CORRELATION1000MATRIX('**incolumn=sepallength:X; sepalwidth:X; petallength:Y; petalwidth:Y**, **...**');	No
call nza..CORRELATION500PAIRS('**x=sepallength;sepalwidth, y=petallength;petalwidth**, **...**');	call nza..CORRELATION500PAIRS('**incolumn=sepallength:petallength; sepalwidth:petalwidth**, **...**');	No
call nza..COV('**X=age, Y=wage_per_hour**, **...**');	call nza..COV('**incolumn=age; wage_per_hour**, **...**');	No
call nza..COVARIANCE1000MATRIX('**x=sepallength;sepalwidth, y=petallength;petalwidth**, **...**');	call nza..COVARIANCE1000MATRIX('**incolumn=sepallength:X; sepalwidth:X; petallength:Y; petalwidth:Y**, **...**');	No
call nza..COVARIANCE500PAIRS('**x=sepallength;sepalwidth, y=petallength;petalwidth**, **...**');	call nza..COVARIANCE500PAIRS('**incolumn=sepallength:petallength; sepalwidth:petalwidth**, **...**');	No
call nza..COVARIANCEMATRIX('**X=wage_per_hour;capital_gains, Y=age**, **...**');	call nza..COVARIANCEMATRIX('**incolumn=wage_per_hour:X; capital_gains:X; age:Y**, **...**');	No

Prior Release (old way) (Sample partial call)	Release 2.0 (new way) (Sample partial call)	Backward Compatible? /Notes
call nza..CUMULATIVE('**X=somecol, ...**');	call nza..CUMULATIVE('**incolumn=somecol, ...**');	No
call nza..DENSITY('**X=somecol, ...**');	call nza..DENSITY(' **incolumn=somecol, ...**');	No
call nza..ENTROPY('**X=WORKCLASS, ...**');	call nza..ENTROPY(' **incolumn=WORKCLASS, ...**');	No
call nza..JOINT_ENTROPY('**X=age, Y=wage_per_hour, ...**');	call nza..JOINT_ENTROPY ('**incolumn=age; wage_per_hour, ...**');	No
call nza..MOMENTS('**X=hours_per_week, ...**');	call nza..MOMENTS('**incolumn=hours_per_week, ...**');	No
call nza..MTBNET_GROW('**varlist=SEPALLENGTH; SEPALWIDTH; PETALLENGTH, ...**');	call nza..MTBNET_GROW('**incolumn=SEPALLENGTH; SEPALWIDTH; PETALLENGTH, ...**');	No
call nza..MUTUALINFO('**X=age, Y=wage_per_hour, ...**');	call nza..MUTUALINFO('**incolumn=age; wage_per_hour, ...**');	No
call nza..MWW_TEST('**x=HOURS_PER_WEEK, ...**');	call nza..MWW_TEST('**incolumn=HOURS_PER_WEEK, ...**');	No
call nza..PMML_**DECTREE**(' **...**'); (function deprecated)	call nza..PMML_**MODEL**(' **...**');	Yes
call nza..PMML_**NAIVEBAYES**(' **...**'); (function deprecated)	call nza..PMML_**MODEL**(' **...**');	Yes
call nza..PRINT_**DECTREE**(' **...**'); (function deprecated)	call nza..PRINT_**MODEL**(' **...**');	Yes
call nza..PRINT_**REGTREE**(' **...**'); (function deprecated)	call nza..PRINT_**MODEL**(' **...**');	Yes
call nza..PPOINT('**X=somecol, ...**');	call nza..PPOINT(' **incolumn=somecol, ...**');	No
call nza..SPEARMAN_CORR_S('**X=RESIDUALSUGAR, Y=CHLORIDES, ...**');	call nza..SPEARMAN_CORR_S('**incolumn=RESIDUALSUGAR; CHLORIDES, ...**');	No
call nza..SPEARMAN_CORR('**X=RESIDUALSUGAR, Y=CHLORIDES, ...**');	call nza..SPEARMAN_CORR('**incolumn=RESIDUALSUGAR; CHLORIDES, ...**');	No
call nza..STD_NORM('**transform=S:petallength; L:petallength; N:petallength; U:petallength; C:sepallength/petallength, ...**');	call nza..STD_NORM('**incolumn=petallength:S; petallength:L; petallength:N; petallength:U; sepallength/petal-length:C, ...**');	No

Prior Release (old way) (Sample partial call)	Release 2.0 (new way) (Sample partial call)	Backward Compatible? /Notes
call nza..SUMMARY1000('**varlist=FIXED_ACIDITY; VOLATILE_ACIDITY; CITRIC_ACID; RESIDUALSUGAR**, **...**');	call nza..SUMMARY1000('**incolumn=FIXED_ACIDITY; VOLATILE_ACIDITY; CITRIC_ACID; RESIDUALSUGAR**, **...**');	No
call nza..T_LS_TEST ('**X=petallength, Y=sepallength**, **...**');	call nza..T_LS_TEST('**incolumn=petallength:X; sepallength:Y**, **...**);	No
call nza..T_ME_TEST ('**X=petallength**, **...**');	call nza..T_ME_TEST('**incolumn=petallength**, **...**');	No
call nza..T_PMD_TEST('**X=petallength, Y=sepallength**, **...**');	call nza..T_PMD_TEST('**incolumn=petallength:X; sepallength:Y**, **...**');	No
call nza..T_UMD_TEST ('**X=sepalwidth, class=class, class1="virginica", class2="versicolor"**, **...**');	call nza..T_UMD_TEST('**incolumn=petallength, class=class:"virginica":"setosa"**, **...** ');	No
call nza..TBNET_GROW('**varlist=SEPALLENGTH; SEPALWIDTH; PETALLENGTH; PETALWIDTH**, **...**');	call nza..TBNET_GROW('**incolumn=SEPALLENGTH; SEPALWIDTH; PETALLENGTH; PETALWIDTH**, **...**');	Yes
call nza..TBNET1G('**varlist=SEPALLENGTH; SEPALWIDTH; PETALLENGTH;PETALWIDTH**, **...**');	call nza..TBNET1G('**incolumn=SEPALLENGTH; SEPALWIDTH; PETALLENGTH;PETALWIDTH**, **...**');	No
call nza..TBNET1G2P('**varlist1=SEPALLENGTH; SEPALWIDTH, varlist2=PETALLENGTH; PETALWIDTH**, **...**');	call nza..TBNET1G2P('**incolumn=SEPALLENGTH:X; SEPALWIDTH:X; PETALLENGTH:Y; PETALWIDTH:Y**, **...**');	No
call nza..TBNET2G('**varlist1=SEPALLENGTH; SEPALWIDTH, varlist2=PETALLENGTH; PETALWIDTH**, **...**');	call nza..TBNET2G('**incolumn=SEPALLENGTH:X; SEPALWIDTH:X; PETALLENGTH:Y; PETALWIDTH:Y**, **...**');	No
call nza..WILCOXON_TEST('**X=VOLATILE_ACIDITY, Y=CITRIC_ACID**, **...**');	call nza..WILCOXON_TEST('**incolumn=VOLATILE_ACIDITY; CITRIC_ACID**, **...**');	No

For general rules regarding the call interface for analytic algorithms, see “Call Interface” in the *IBM SPSS In-Database Analytics Developer's Guide*. For complete usage information, see the *IBM SPSS In-Database Analytics Reference Guide*.

FPGROWTH Algorithms Renamed and Modified

The FPGROWTH algorithm has been renamed to ARULE and some parameters have changed. There is no backward compatibility; the FPGROWTH algorithms are no longer

supported. The table below shows sample calls for both the older and new algorithm names.

Prior Release (old way)	Release 2.0 (new way)
CALL nza..**PREPARE_FPGROWTH**('intable=nza..quant_sales, outtable=dset, tid=tid, item=idart');	CALL nza..**PREPARE_ARULE**('intable=nza..quant_sales, outtable=dset, tid=tid, item=idart');
CALL nza..**FPGROWTH**('intable=nza..retail, **pfx=results**, support=1 ');	CALL nza..**ARULE**('intable=nza..retail, **model=assoc**, **supporttype=percent**, support=5, **lvl=0, maxsetsize=5, confidence=0.5**');

Detailed information about these algorithms can be found in the *IBM SPSS In-Database Analytics Developer's Guide* and the *IBM SPSS In-Database Analytics Reference Guide*.

Modified Default Parameter Settings for DECTREE and REGTREE

Default parameter settings for decision and regression tree growing are modified in this release to yield smaller trees. The previous defaults resulted in creating large trees, maximally fitted to the training data (good candidates for pruning). Growing, pruning, and using such large trees for prediction is computationally expensive and often unnecessary. By using modified parameter settings, you can obtain substantial computational savings. The algorithms are usually run with their default settings, so those defaults are changed to produce smaller trees. To force large tree growing, those users with an understanding of the performance impact can change the parameter settings.

Decision Trees

The following default parameter settings are changed:

Parameter	New Value	Old Value	Description
minsplit	50	2	The minimum number of instances in a node required for a split. If the number of instances in a node is less than minsplit, no further split is applied and the node becomes a leaf.
maxdepth	10	62	The maximum decision tree depth. If a node's level in the tree equals maxdepth (with the level of the root node equal 1, the level of its descendants equal 2, etc.), no further split is applied and the node becomes a leaf.
minimprove	0.02	0.01	The minimum improvement of the evaluation function used for split selection (class impurity measure) required for a split. If the decrease of class impurity (calculated using the function specified via the eval argument, defaulting to the entropy) is less than minimprove, no further split is applied and the node becomes a leaf.

Regression Trees

The following default parameter setting is changed:

Parameter	New Value	Old Value	Description
minsplit	50	2	The minimum number of instances in a node required for a split. If the number of instances in a node is less than minsplit, no further split is applied and the node becomes a leaf.

Time Series Forecasting

Support for Time Series is introduced in this release. A time series is a sequence of numerical data values, measured at successive, but not necessarily equidistant points in time. Examples are daily stock prices, monthly unemployment counts, or annual changes in global temperature. The two main goals of time series analysis are to understand the underlying patterns that are represented by the observed data and to make forecasts. Time Series support is implemented with the following new algorithm:

TIMESERIES (NVARCHAR(ANY) paramString)

Detailed information about this algorithm can be found in the “Time Series Forecasting” section of the *IBM SPSS In-Database Analytics Developer's Guide*.

Missing Value Support in Analytics Functions

In prior releases of Netezza Analytics, analytic algorithms were unable to work with tables that were missing values in the columns being used in the algorithm's calculation. Because many real world databases suffer from missing values in tables, preprocessing was required in these cases to either remove rows or columns with missing values, replace missing values with some special value, or to impute the value by using the Netezza Analytics supplied IMPUTE_DATA procedure. New to this release is an internal solution built into various algorithms to deal with the missing values. This provides:

- A more convenient solution
- Possibly better model quality
- Possibly better predictions

The following selected algorithms are capable of building or applying models using tables with missing values, internally handling missing values in an appropriate manner (instead of just ignoring instances with missing values):

- Decision Trees
- Regression Trees
- Naïve Bayes classifier

For other algorithms, if rows contain missing values, the rows are ignored, but the table is still used. Preprocessing is still possible, using the Netezza Analytics supplied IMPUTE_DATA procedure, but is not required. Note that preprocessing is not “automated.” Detailed information about how missing values are handled can be found in the *IBM SPSS In-Database Analytics Developer's Guide*.

Changes to the KMEANS Algorithm

The following new features were added to the existing KMEANS algorithm:

- Clustering using Mahalanobis distance
- Normalized Euclidean distance
- Scoring with statistics of clusters and columns
- Automatic data normalization and standardization
- Enriched statistics

See “KMEANS algorithm” and “Enriched Statistics for Clustering Models” in the *IBM SPSS In-Database Analytics Developer's Guide* for details of these new features.

In this release there is a behavior change to the KMEANS algorithm. By default, the 'normalized Euclidean distance' is used by the KMEANS algorithm if the distance option is not specified. In prior releases, 'Euclidean distance' is the default distance used.

Metadata Management For Analytic Models

The primary goal of the new Metadata Management feature is to provide an environment for managing the analytic models created by the Netezza Analytics software. The implementation of the Metadata Management component is done on top of the existing database system, using stored procedures and user-defined functions.

All analytics models created by the various Netezza Analytics functions (like DECTREE or KMEANS) are registered in a catalog, and new administrative and other functions are offered for model management. The Metadata Management system provides the following features:

- List information about models
- Perform basic operations on models (for example, delete, copy, rename, update)
- Perform advanced operations on models (for example print, PMML format, export)
- Security (grant and revoke privileges on models and model operations)

Note that this new feature is required by all algorithms that generate models. When you enable a database for Netezza Analytics using the script **create_inza_db.sh**, the database is automatically prepared for the Metadata Management feature.

This new feature is described in more details in the “Metadata Management” section of the *IBM SPSS In-Database Analytics Developer's Guide*.

Most models created using prior releases of Netezza Analytics can be registered in the metadata catalog so that they can be used with Metadata Management. If model migration is needed, it is done automatically. See the REGISTER_MODEL procedure in the *IBM SPSS In-Database Analytics Developer's Guide*.

Limited PMML Support for Analytic Models

PMML (Predictive Model Markup Language) is defined by the Data Mining Group (DMG) and is the widely accepted standard for the exchange of data mining models. Limited PMML support is provided in this version. Support will be producer conformance for decision tree (classification), association rules, naïve Bayes, and *k*-means (clustering) models. PMML support enables users to employ PMML-conformant model visualization tools, such as the InfoSphere Warehouse visualizer, for model exploration. It also allows scoring of Netezza Analytics models in DB2. The following algorithms have limited PMML support:

- Decision trees

- *K*-means
- Association rules (ARULE)
- Naïve Bayes

Support is implemented with the following new analytic procedures:

- PMML_MODEL
- EXPORT_PMML

This new feature is described in more details in the “PMML” section of the *IBM SPSS In-Database Analytics Developer's Guide*.

Logistic Regression and Generalized Linear Models (GLM)

New to this release are algorithmic procedures to support GLM. These are:

- GLM
- PREDICT_GLM

Detailed information on these new procedures can be found in the “Generalized Linear Models” section of the *IBM SPSS In-Database Analytics Developer's Guide*.

Netezza Lua API for Analytics

Beginning with Release 2.0, Netezza Analytics includes a variant of Lua, which enables users to take advantage of the Netezza appliance's highly parallelized environment. Lua is an extension programming language designed to support general procedural programming with data description facilities. It also offers good support for object-oriented programming, functional programming, and data-driven programming. Lua is intended to be used as a powerful, lightweight scripting language for any program that needs one. For more information on Netezza Lua, see the *IBM Netezza Lua Developer's Guide*.

Perl Adapter For User-Defined Analytic Procedures

Users can extend SQL with user-defined functions (UDXs) as well as user-defined analytic processes (UDAPs). The UDAP concept allows a user to implement a freestanding, executable data-processing program that runs “out of process” (that is, outside the database system), and register it in a database. UDAPs provide several important features that are not available with a UDX. For example, a UDX must be coded in C++ but UDAPs can be written in a variety of languages, allowing users to capitalize on their existing programming skills. Release 2.0 provides support for writing UDAPs in the Perl language. For more information on UDAPs and the Perl Adapter, see the *User-Defined Analytics Process Developer's Guide*.

Changes to nzMatrix

nzMatrix has improved the matrix multiplication procedure (GEMM) and introduced new random number generators in this release.

Simplified Matrix Multiplication

The matrix multiplication procedure (GEMM) has been simplified. Previously, users chose whether to use GEMM or GEMM_LARGE, based on speed requirements and matrix size. (GEMM was faster but could not calculate larger matrices.) **With this release, the GEMM procedure has been enhanced and GEMM_LARGE is no longer required.** Beginning with this release, GEMM uses the following signatures (the second being previously part of GEMM_LARGE):

- GEMM (nvarchar(any), nvarchar(any), nvarchar(any))
- GEMM (nvarchar(any), BOOLEAN, nvarchar(any), BOOLEAN, nvarchar(any)))

The system selects the appropriate algorithm, transparent to the user, based on the size of the processed matrices and architecture of the hardware. Note that, for backward compatibility, the system continues to recognize the GEMM_LARGE stored procedure.

New Random Number Generators

This Netezza Analytics release introduces a new set of wrappers on the Intel Math Kernel Library® random number generators (RNGs). The API provides a set of stored procedures that generate matrices filled with random values. Following are the new stored procedures:

Stored Procedure	Description
CREATE_RANDOM_CAUCHY_MATRIX	Create a random matrix using Cauchy distributed random values
CREATE_RANDOM_EXPONENT_MATRIX	Create a random matrix using Exponential distributed random values
CREATE_RANDOM_GAMMA_MATRIX	Create a matrix of pseudo-random variables following the Gamma distribution
CREATE_RANDOM_LAPLACE_MATRIX	Create a matrix of pseudo-random variables following the Laplace distribution
CREATE_RANDOM_NORMAL_MATRIX	Create a matrix of pseudo-random variables following the normal distribution
CREATE_RANDOM_POISSON_MATRIX	Create a matrix of pseudo-random variables following the Poisson distribution
CREATE_RANDOM_RAYLEIGH_MATRIX	Create a Matrix of random using a Rayleigh distributed random values generator
CREATE_RANDOM_UNIFORM_MATRIX	Create a matrix of pseudo-random variables following the uniform distribution
CREATE_RANDOM_WEIBULL_MATRIX	Create a matrix of pseudo-random variables following the Weibull distribution

See *Intel Math Kernel Library Vector Statistical Library Notes* for more information:

http://software.intel.com/sites/products/documentation/hpc/mkl/vslnotes/vslnotes.pdf

Changes to Netezza Spatial

Spatial precision has changed such that coordinate values display only the value's significant digits up to fifteen digits of precision which is the maximum for 64-bit floating point values. Prior to 2.0, by default, the user would always see 16 decimal digits. This means that any trailing 0's at the end of a value will now be truncated in this release. For example, prior to this release the user would have seen the following:

```
INZA(ADMIN)=> select st_astext(st_wkttosql('point (10 16)'));
                  ST_ASTEXT
-----------------------------------------------
 POINT (10.0000000000000000 16.0000000000000000)
(1 row)
```

With 2.0 they will now see:

```
INZA(ADMIN)=> select st_astext(st_wkttosql('point (10 16)'));
   ST_ASTEXT
---------------
 POINT (10 16)
(1 row)
```

Additionally, prior to this release, the user could control the number of decimal digits as follows:

```
INZA(ADMIN)=> select st_astext(st_wkttosql('point (10 16)'), 8);
            ST_ASTEXT
---------------------------------
 POINT (10.00000000 16.00000000)
(1 row)
```

With this release this option is no longer available.

Issues Fixed In Release 2.0

Reference	Topic/Area	Issue Description
EXT-1084	PCA	Performance Improvements
EXT-1509	ARULE (formerly FPGROWTH)	Performance Improvements
EXT-1518	Netezza Matrix Engine	When using RCV2SIMPLE_NUM or RCV2SIMPLE to convert a row/column/value table to a “simple” matrix table may fail if the number of projected columns is greater than 1600.
EXT-1591	DECTREE	Performance Improvements for large datasets.
EXT-1647	Algorithm	MSE,CAE, RAE, RSE return null instead of an error when column does not contain unique value.
EXT-1688	DECTREE	Performance Improvements. Full class distribution enhances model inspection possibilities and may improve probabilistic prediction quality.
EXT-2097	Matrix	Previous version of CREATE_MATRIX_FROM_TABLE introduced a decrease of performance for valid, dense input tables. This fix improves performance for DENSE input data.
EXT-2113	Matrix	Performance Improvement to nzm..CONCAT procedure.

Known Issues in Release 2.0

The following are known issues in Release 2.0. Those references numbers shown in red have been fixed in a later patch release.

Reference	Topic/Area	Issue Description / Workaround

	Netezza Matrix Engine	Using CTRL-C in nzsql typically aborts and rolls back the transaction in progress. However, it is possible that the Matrix Engine processes continue running and consuming resources. To check if any Matrix Engine processes are running, use the following SQL query:
		`CALL NZA..SP_MPI_STATS();`
		
		To abort a Matrix Engine, use the following SQL query (replacing "123456789" with the engine's job ID):
		`CALL NZM..KILL_ENGINE(123456789);`

	Netezza Matrix Engine	Calculations using PBLAS or ScaLAPACK consume S-Blade RAM for storing input matrices, intermediate work matrices, and result matrices. Each matrix element consumes 8 bytes. Exceeding available RAM may result in S-Blade restarts and aborted computations. Available RAM equals total RAM minus the RAM requirements of the Linux operating system, the Netezza system, and concurrent, unrelated queries. When using matrices please keep these maximum numbers in mind. In future version more controls will be implemented to help guard against exceeding these limits.
	Netezza Matrix Engine	An issue can exist if the matrix engine attempts to take up too much memory on the Netezza system (if the size of the matrix and its calculations becomes too large). From a user's perspective, you may experience two situations: ▶ The matrix operation returns with "Out of memory" exception when the internal MatrixEngine assertion fails. ▶ Matrix operation returns with a "timeout" error when it is killed by our OOM guard.
	Netezza Matrix Engine	The Netezza Matrix Engine uses the double precision (64-bit) floating point approximate numeric data type for storage and computation of matrix element values. Row and column indices are stored as 32-bit integer values, allowing up to 2,147,483,647 rows and columns. Row and column indices begin at 1. An index value of zero is not permitted.
EXT-836	R Analytic Executables	Depending on the data types used for records, transfer of data from the table to R memory can be very slow.
EXT-951	QUANTILE_DISC	Use of the QUANTILE_DISC function may cause a core dump.

Reference	Topic/Area	Issue Description / Workaround

EXT-1107	Netezza Spatial Package	A ST_DWithin function performed on two points does not return TRUE when increasing the distance value past 18945535.
		`nzsql -u $NZ_USER -pw $NZ_PASSWORD -c "SELECT ST_DWithin(ST_Point(0,0), ST_Point(0,1), 18945535, 'meter', 'wgs84');"`
		` ST_DWITHIN`
		`------------`
		` t`
		`(1 row)`
		
		`nzsql -u $NZ_USER -pw $NZ_PASSWORD -c "SELECT ST_DWithin(ST_Point(0,0), ST_Point(0,2), 18945536, 'meter', 'wgs84');"`
		` ST_DWITHIN`
		`------------`
		` f`
		`(1 row)`

EXT-1223	Netezza Matrix Engine	All matrices in a database are owned by INZAUSER. Also, any "inza user" in the database runs the Matrix Engine commands as INZAUSER. Therefore, if user A creates a matrix, then user B can see the matrix. It is not possible to grant that permission (or revoke it) to make matrices invisible to users. This is in conflict with all other database objects in a Netezza system. Objects have permissions granted on them (or revoked on them) to users - otherwise the users cannot see them or operate on them. Furthermore, any Netezza Analytics user in a database can drop any other user's matrices. There is no protection.

EXT-1224	Netezza Matrix Engine	(Related to EXT-1223)
		Users who have not been given permissions to a particular database on the Netezza System may still be able to list, view, print, and delete matrices on that database, despite not being able to affect tables. For example:
		`CALL nzm..list_matrices();`
		`CALL nzm..print('ones5x5');`
		`CALL nzm..delete_matrix('ones5x5');`

EXT-1225	Netezza Matrix Engine	If a user is granted Netezza Analytics permissions in a particular database, then that user can use Netezza Analytics functions in any other database where they have user permissions (except nzMatrix functions). Note that when you grant a user access to Netezza Analytics, you are performing a global operation, not a per-database operation, with the exception of nzMatrix which is database-local. While working as designed, improper use may cause issues.
EXT-1248	KNN, KMEANS	DATE type is not supported by KNN and KMEANS algorithms
EXT-1249	MUTUALINFO	The MUTUALINFO procedure may hang when run on a IBM Netezza 1000-12.
EXT-1255	Netezza Analytics	Moments, correlations, covariance, Bayesian networks, and standardization/normalization stored procedures do not allow NULL values. To use these stored procedures, all NULL values must be removed from the data.

Reference	Topic/Area	Issue Description / Workaround
EXT-1413	Netezza Analytics	Some objects in the database are owned by ADMIN instead of INZAUSER, which may cause access issues.
EXT-1593	DECTREE, REGTREE	Tree scoring has a non linear scoring curve.
EXT-1801	TBNET1G2P	Algorithm TBNET1G2P does not support column properties in metadata management tables
EXT-1928	System	Cron job which cleans remote analytic processes is launched when there are previous ones still running.
EXT-1955	MTBNET_GROW	Algorithm MTBNET_GROW does not support column properties in metadata management tables
EXT-1956	TBNET_GROW	Algorithm TBNET_GROW does not support column properties in metadata management tables
EXT-1957	TBNET1G	Algorithm TBNET1G does not support column properties in metadata management tables
EXT-1959	TBNET2G	Algorithm TBNET2G does not support column properties in metadata management tables
EXT-1983	KMEANS	KMeans calling nza..PRINT_MODEL returns an error PRINT_MODEL function is not supported for this type of model.
EXT-1988	Time Series	Time series, STD does not find the sinus when it is not adapted to the stepsize.
EXT-1991 and SWS-74183		Sporadically, when running certain Netezza Analytics algorithms the following error may occur: `ERROR: 3 : Expected bool return type` This has been noted with naïve bayes algorithms and kmeans. This is a Netezza system error which will be fixed in an upcoming release. Contact Customer Support for a more up to date status on a patch release to fix this issue.
EXT-2052	User Defined Analytic Executable	Problem with NULL field handling in C++ User Defined Analytic Executable
EXT-2075	Time Series	Forecasttimes sometimes only returns the last requested time.
EXT-2177	Time Series	Timeseries, if there are no periods FFT seems to discover periods in just noise
EXT-2179	Time Series	Initialization of Seasonal Coefficients for Exponential Smoothing should use detrended data.

Reference	Topic/Area	Issue Description / Workaround
EXT-2209 and EXT-2101	Metadata Management	nzconvertsyscase does not work in conjunction with Netezza Analytics. Conversion of the system case from uppercase to lowercase or vice versa using the command "nzconvertsyscase" does not convert the metadata management tables. Before you run this command, you must (for all databases) drop all analytics models and use the nza..cleanup() procedure to remove the metadata. After you run the command, you must re-register the Netezza Analytics component analytics_utils: nzcm -fr analytics_utils

EXT-2209 and SWS-72806 and SWS-74893	Metadata Management	There are two issues related to backup/restore operations with regard to Metadata Management.
		
		1. When you backup a database that contains Netezza Analytics models and restore it on ANOTHER Netezza system, the table/view references in the metadata tables could be incorrect. If you need to do so, the administrator should set the object ids used in the metadata tables to the value -1 (for all databases containing analytics models) after the restore operation: In table NZA_META_MODELS, these are the columns OWNERID and CREATORID; and in table NZA_META_COMPONENTS, the column OBJID.
		2. Restore of a database containing metadata management views may fail due to issues SWS-72806 and SWS-74893. This is a NON-FATAL error; the database and all the data in the tables will be restored correctly, but some views MAY be missing. These views can be manually re-created by doing the following:
		• Log in as ADMIN
		• Find the Model IDs for all models that have a column properties view registered in MM:
		`select id from nza_meta_components where usagetype = 'Column Properties';`
		• For all these IDs (here with ID 29 as example), run the following two internal procedures:
		`call nza..sp_meta_remove_component( 29, 'Column Properties', -1);`
		`call nza..sp_meta_create_column_property_view( 29, 'COLNAME', false);`

EXT-2255	STD_NORM	STD_NORM returns columns with null values even though standardization and normalization is feasible. This issue that may indirectly impact KMEANS auto-transformation since KMEANS automatic standardization or normalization may returns columns with null values when columns have very small variance values.
EXT-2318	TBNET_APPLY	The output table is different in this release. The prediction column is named <colname>_PRED instead of <colname>_pred.

Reference	Topic/Area	Issue Description / Workaround
EXT-2319	nzSpatial	ST_PointOnSurface incorrectly returning a point with empty polygons
EXT-2328	Bayesian Networks	Bayesian Networks are not deterministic in the choice of VARX and VARY
EXT-2332	KMEANS	The number of numeric columns supported are limited to 55 in this release when Mahalanobis distance is used for K-means clustering.
SWS-62976, EXT-915	Netezza system, Netezza Analytics	A known issue has surfaced in recent testing regarding algorithms calling the function rank() or over(). When these functions are called on large data sets (10 billion rows if called once, or 5 billion rows if called twice), you will receive an error similar to, "ERROR: SPU swap partition: Disk temporary work space is full." This is due to the function operating on host where there is not enough disk space to handle the operation. To solve this issues properly, a Netezza system fix is needed (SWS-62976). This is currently slated to be included in a patch release for Netezza systems (date to be determined). Until the fix is in place, you are limited to a data set less than 5 or 10 billion rows as noted above. The following Algorithms use either rank() or over(): **rank()** ▶ Correlation and Mutual Information ▶ Discretization ▶ FP-Growth ▶ General Diagnostic Measures ▶ KNN ▶ Moments ▶ Non-Parametric Statistics ▶ PCA added to the Analytics documentation **over()** ▶ Correlation and Mutual Information ▶ Decision Trees ▶ Discretization ▶ Divisive Clustering ▶ FP-Growth ▶ General Diagnostic Measures ▶ *K*-means ▶ KNN ▶ Moments ▶ Non-Parametric Statistics ▶ Regression Trees ▶ Sufficient Statistics

Netezza Analytics Release 2.0.1

The Netezza Analytics Release 2.0.1 patch release contains bug fixes and improvements to the documentation.

Call Interface Changes

The call interface to the following analytic function has changed in this release:

Prior Release (old way) (Sample partial call)	**Release 2.0 (new way)** (Sample partial call)	**Backward Compatible?**
call nza..BITABLE('**incolumn1=income; incolumn2=education ...**')	call nza..BITABLE(' '**incolumn=income:x; education:y, …**')	No

Issues Fixed In Release 2.0.1

The following issues were fixed in this release:

Reference	Topic/Area	Issue Description
EXT-1248	KNN, KMEANS	DATE type working properly when date columns are treated as nominal.
EXT-2318	TBNET_APPLY	The outtable column name case is no longer an issue.
EXT-2340	nzTApply	The nzTApply function now properly passes column names when machine is set to lowercase.
EXT-2355	Lua	nzLua now has an API call for the getCurrentUsername() UDX function, allowing a UDX to get the name of the user executing the UDX.
EXT-2357	timetz literal value	The parameters forecasthorizon and forecasttimes now accept a colon in the timetz format.
EXT-2361	KMEANS	Calculation of cardinality is now correct when 'statistics = values'.
EXT-2380	Lua	There is now a check for 0 prior to division and there is no longer a subtraction of 1 from the ending value.
EXT-2418	Installation	Registration error during installation has been fixed.
EXT-2419	Linear Regression	The t-statistics and p-values for significance of the regression coefficients, as well as the R^2 coefficient, are now computed correctly.

Documentation Changes

Release 2.0.1 contains the following two manuals documenting the Netezza Analytics map/reduce functionality.

- IBM Netezza Analytics Map/Reduce API Reference
- IBM Netezza Analytics Map/Reduce Developer's Guide

Notices and Trademarks

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
26 Forest Street
Marlborough, MA 01752 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only. This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. (enter the year or years). All rights reserved.

Trademarks

IBM, the IBM logo, ibm.com and Netezza are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or com-

mon law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at ibm.com/legal/copytrade.shtml.

The following terms are trademarks or registered trademarks of other companies:

Adobe is a registered trademark of Adobe Systems Incorporated in the United States, and/or other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

NEC is a registered trademark of NEC Corporation.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Red Hat is a trademark or registered trademark of Red Hat, Inc. in the United States and/or other countries.

D-CC, D-C++, Diab+, FastJ, pSOS+, SingleStep, Tornado, VxWorks, Wind River, and the Wind River logo are trademarks, registered trademarks, or service marks of Wind River Systems, Inc. Tornado patent pending.

APC and the APC logo are trademarks or registered trademarks of American Power Conversion Corporation.

Other company, product or service names may be trademarks or service marks of others.

Regulatory and Compliance

Regulatory Notices

Install the Netezza system in a restricted-access location. Ensure that only those trained to operate or service the equipment have physical access to it. Install each AC power outlet near the Netezza system rack that plugs into it, and keep it freely accessible. Provide approved circuit breakers on all power sources.

Product may be powered by redundant power sources. Disconnect ALL power sources before servicing. High leakage current. Earth connection essential before connecting supply. Courant de fuite élevé. Raccordement à la terre indispensable avant le raccordement au réseau.

Homologation Statement

Attention: This product is not intended to be connected directly or indirectly by any means whatsoever to interfaces of public telecommunications networks, neither to be used in a Public Services Network.

FCC - Industry Canada Statement

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial

environment. This equipment generates, uses, and can radiate radio-frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case users will be required to correct the interference at their own expense.

This Class A digital apparatus meets all requirements of the Canadian Interference-Causing Equipment Regulations.

Cet appareil numérique de la classe A respecte toutes les exigences du Règlement sur le matériel brouilleur du Canada.

WEEE

Netezza Corporation is committed to meeting the requirements of the European Union (EU) Waste Electrical and Electronic Equipment (WEEE) Directive. This Directive requires producers of electrical and electronic equipment to finance the takeback, for reuse or recycling, of their products placed on the EU market after August 13, 2005.

CE Statement (Europe)

This product complies with the European Low Voltage Directive 73/23/EEC and EMC Directive 89/336/EEC as amended by European Directive 93/68/EEC.

Warning: This is a class A product. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.

VCCI Statement

この装置は、情報処理装置等電波障害自主規制協議会 (VCCI) の基準に基づくクラス A 情報技術装置です。この装置を家庭環境で使用すると電波妨害を引き起越すことがあります。この場合には使用者が適切な対策を講ずるう要求されることがあります。